Maximum Clique in a Graph

BACKGROUND OF THE INVENTION

Technical Field

This invention relates to a method and system for maximizing connectivity between each node among a grouping of nodes in a network computing environment through maximizing connectivity for each member of a clique in a graph. More specifically, the invention relates to systematic removal of vertices in a graph that have an inefficient connectivity count.

Description Of The Prior Art

A connectivity count is a mathematical relationship illustrating interconnections between objects in a group. In a computing environment, connectivity between server nodes in a cluster enhances communication and operating efficiency of the cluster. Total connectivity among nodes is obtained when each node in a grouping of nodes is connected to each other node in the grouping. This grouping is known as a clique. In order to maintain an efficient operating cluster, it may be desirable to remove a node from the cluster that it not completely connected to each other node in the cluster.

There are several known methods for determining connectivity among nodes in a cluster. One known method is to determine connectivity through a build approach. Fig. 1 is a flow chart (10) illustrating a generic build approach algorithm. The process is initiated with computing a connectivity count of all the vertices in the graph (12). Thereafter, the vertices in the graph are sorted (14), and a clique set within the graph is initialized as a null set (16). A test is subsequently conducted to determine if the graph is empty (18). If the result of the test at step (18) is positive, the clique is returned (20).

However, if the result of the test at step (18) is negative, the vertex with the highest connectivity is selected and removed from the graph (22). Thereafter, another test is conducted to determine if the removed vertex is connected with all of the vertices in the graph (24). A positive response to the test at step (24), will result in adding the vertex to the graph (26), followed by a return to step (18). Alternatively, a negative response to the test at step (24), will result in a return to step (18). The build algorithm, as demonstrated in Fig. 1, is initiated with an empty list of nodes. A first node is selected, and a search is conducted to determine which other nodes are connected to the first node selected. This approach is continued for each node in the cluster. A graph is built based upon the connectivity data collected for each node, thereby allowing the operator to determine connectivity for each node in the computing environment. The build approach iteratively adds nodes to build a clique with maximum connectivity among the nodes. One limitation associated with the build approach is the time constraint of determining connectivity for each node in the cluster on an individual basis. Accordingly, the build approach is a deferred algorithm for determining connectivity among nodes in a cluster.

There is therefore a need for an efficient method and system to determine connectivity among peer nodes in a cluster.

SUMMARY OF THE INVENTION

This invention comprises a method and system for efficiently determining connectivity among vertices in a graph.

In a first aspect of the invention, a method is provided for maximizing group membership. A connectivity count of each vertex in a graph is calculated, and a maximum connectivity count for each vertex is determined based upon the calculation. A vertex with a connectivity count less than the maximum connectivity count is removed from the graph.

In a second aspect of the invention, a system is provided to determine a maximum group membership. A counter is provided for calculating a connnectivity count for each vertex in a graph. Means are provided for placement of each vertex in descending order of connectivity, and means are provided for removal of a vertex from the graph with a connectivity count less than a maximum connectivity count.

In a third aspect of the invention, an article is provided with a computer-readable signal-bearing medium. Means in the medium are provided for calculating connectivity for each vertex in a graph. In addition, means in the medium are provided for selecting a least connected vertex in a clique in the graph, and for removing the least connected vertex from the graph.

Other features and advantages of this invention will become apparent from the following detailed description of the presently preferred embodiment of the invention, taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a prior art flow diagram for achieving maximum connectivity.

FIG. 2 is a flow diagram of an algorithm for achieving maximum connectivity according to the preferred embodiment of this invention, and is suggested for printing on the first page of the issued patent.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Overview

In a multi-node computing environment, inter-connectivity among nodes in a cluster enhances coordination of communication with other nodes in the environment. Maximum connectivity is achieved when two-way communication exists between each node in the cluster. Each node in the cluster is mapped on a graph illustrating the number of vertices associated with each node, wherein each vertex is representative of the number of nodes in the cluster to which each node is connected. Vertices which are not in communication with each vertex in the graph are eliminated in an expedited manner. Accordingly, the process of eliminating least connected vertices results in an efficient algorithm for returning a maximum clique of vertices.

Technical Details

Fig. 2 is a flow diagram (30) illustrating the process of removal of vertices from a graph. The first step in the process is to calculate a connectivity count for each vertex in the graph (32). A connectivity count is a mathematical relationship illustrating interconnections between vertices. Following the step of calculating a connectivity count, each of the vertices of the graph is placed in decreasing order of connectivity (34). This placement process is a component that is required for the subsequent selection process. Thereafter, a test is conducted to determine if the connectivity count of a least connected vertex is equal to the quantity of vertices in the graph (36). A positive response to the test at step (36) is an indication that a maximum clique may have been found. Thereafter, a subsequent test is conducted to determine if the remaining vertices in the graph form a clique, and if the size of the clique is larger than any previously noted clique (38). A positive response to the test at step (38) will return the clique found at step 36

with the maximum number of vertices in the graph (40). A negative response to the test at step (38) will return a previously noted clique (42). For example, a return of a clique within a multi-node computing system at step (42) signifies a clique of interconnected nodes that is smaller than a previously noted clique. Accordingly, when the connectivity count of the least connected vertex is equal to the number of vertices on the graph this may be an indication that a clique of vertices with maximum connectivity has been achieved.

If at step (36), it is determined that the connectivity count of the least connected vertex is not equal to the number of vertices in the graph , a selection of a vertex from the graph among the least connected vertices is conducted such that the sum of the connectivity count is the least among the vertices in the graph (44). If there is more than one vertex of the least connected vertices with the same connectivity count, the algorithm will select the vertex which when removed will affect the connectivity of the least connected vertices. The selected vertex is removed from the graph (46). Following removal of the selected vertex at step (46), an update of the connectivity count of all of the removed vertex's neighbors is conducted (48). Thereafter, a test is conducted to determine if the removed vertex had a connectivity count of zero prior to deletion from the graph (50). If no other vertex in the graph was connected to the removed vertex, this is an implication that the removed vertex was part of a clique of vertices previously removed. A negative response to the test at step (50) will return to step (34) for placement of the remaining vertices of the graph in descending order of their connectivity. Alternatively, a positive response to the test at step (50) will require a notation of the vertex removed at step (46) together with all of the vertices that were removed from the graph in previous iterations whose connectivity count at the time of deletion was one more than the vertex removed in the previous iteration (52). All of the vertices in this notation at step (52) form a clique. The size of this clique is noted. Thereafter, a test is conducted to determine if the size of the clique is larger than any previously noted cliques of vertices (54). A negative response to the test at step (54)

will result in a return to step (34). However, a positive response to the test at step (54) will result in a notation of the number of vertices in the new clique and the new clique, and erasing the information pertaining to the previous clique (56), followed by a return to step (34). Accordingly, if the connectivity count of the least connected vertex is not equal to the number of vertices in the graph, this vertex is eliminated from the graph.

Vertices in a clique get eliminated from the graph in consecutive iterations. The connectivity count of the vertex at the time of elimination is always one less than it's clique neighbor that was eliminated in the previous iteration. In a systematic and efficient manner, a clique with maximum connectivity is determined. Accordingly, the algorithm proceeds in an accelerated manner to achieve maximum inter-connectivity among vertices in a graph.

Advantages Over The Prior Art

The algorithm disclosed herein analytically solves a geometric clique problem in polynomial time. Vertices in the graph are eliminated from the graph on an individual basis until a completely inter-connected clique of vertices remains. The efficiency of the algorithm allows for a return of a maximum clique of vertices in an expeditious manner.

Alternative Embodiments

It will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without departing from the spirit and scope of the invention. In particular, the vertices of the graph should not be limited to server nodes in a computer system. Rather the vertices of the graph may represent components on an electronic circuit board wherein each node in the graph represents a component, and an edge represents two nodes that are physically too close to be checked simultaneously. A clique in this graph is then a set of components that can be checked

in one pass. Another example of application of the algorithm is for use in pattern recognition. Given a target picture and an input picture (which may involve only a set of points), a related compatibility graph is created whose vertices correspond to pairs of points. There is an edge between two vertices if the corresponding pairs are mutually consistent, where this can depend on a variety of restrictions, including angular relationships as well as the requirement that no point be matched with more than one other. A large clique represents a large number of mutually consistent pairs, and its size can be used as a measure of the corresponding fit. Another example of an application of the algorithm is for use with analysis of biological and archeological data. In biology and archeology, a standard model for relating objects is that of a tree representing the division of a species into two separate species or the division of features of some artifact. A graph may be created wherein the nodes of the graph represent partitions of items. A clique in this graph may represent a set of partitions that can be formed into a tree. Maximum cliques attempt to encapsulate as much of the partition data as possible. Other application of the maximum clique algorithm include project selection, classification, fault tolerance, coding, computer vision, economics, information retrieval, signal transmission, alignment of DNA with protein sequences, and any system where maximum inter-connectivity of all elements in a set is desired. Accordingly, the scope of protection of this invention is limited only by the following claims and their equivalents.